FAST LONG-TERM MOTION ESTIMATION FOR HIGH DEFINITION VIDEO SEQUENCES BASED ON SPATIO-TEMPORAL TUBES AND USING THE NELDER-MEAD SIMPLEX ALGORITHM

Olivier Brouard, Fabrice Delannay, Vincent Ricordel, Dominique Barba

FAST LONG-TERM MOTION ESTIMATION FOR HIGH DEFINITION VIDEO SEQUENCES BASED ON SPATIO-TEMPORAL TUBES AND USING THE NELDER-MEAD SIMPLEX ALGORITHM

Olivier Brouard, Fabrice Delannay, Vincent Ricordel, and Dominique Barba

University of Nantes – IRCCyN laboratory – IVC team
Polytech' Nantes, rue Christian Pauc, 44306 Nantes, France
olivier.brouard@univ-nantes.fr

ABSTRACT

Multi-frame motion estimation is a new method incorporated into the video coding standard H.264/MPEG-4 Advanced Video Coding (AVC) to improve compression performances. Depending on content, the motion estimation can be short or long-term. For long-term motion estimation, the initial search point in the reference frame is important. In this paper[1], we propose a new method for a fast long-term motion estimation with high definition (HD) sequences. We use an implicit uniform motion model. First we describe the multi-resolution motion estimation based on spatio-temporal tubes. These tubes provide a good initial search point for the long-term motion estimation that follows. Then, the search is refined using the Nelder-Mead Simplex method. The global approach reduces computational cost and improves the accuracy of motion estimation.

Index Terms— Long-Term Motion Prediction, Multi-Resolution Motion Estimation, Nelder-Mead Simplex, Spatio-Temporal Tubes.

1. INTRODUCTION

The new video coding standard H.264/MPEG-4 AVC [1] developed by the Joint Video Team of ISO/IEC MPEG and ITU-T Video Coding Expert Group aims at allowing a bit rate reduction of 50% compared to MPEG-2 for the same restitution quality. This higher compression efficiency is obtained using different specifications which are in particular multi-prediction modes, multi-reference frames, and higher motion vector resolution. As a consequence the motion estimation represents the most time consuming part.

In order, to reduce the computational cost of motion estimation, several fast search algorithms have been proposed. Some of them use multi-resolution motion estimation [2][3]. In these methods, frames are first spatially reduced and then motion estimation is computed at the smallest resolution of the frame. The motion vector obtained for lower resolution is then used as motion vector prediction for the higher resolution. This scheme reduces the search window size and accelerates the motion estimation.

The H.264/MPEG-4 AVC encoder and decoder use two lists of previously coded/decoded pictures, which are the reference pictures for the motion estimation. Although, the utilization of multiple reference frames for motion estimation provides significant coding gain [4], computational complexity increases dramatically. The stored frames in the two lists of previously coded/decoded pictures can be short or long-term reference pictures. In a video, the immediately previous frame is often highly correlated with the current frame. Thus, the search window for motion estimation is usually centered on the block of the current frame to predict. However, for long-term reference pictures, contents of the video may can undergo important displacements. With a multi-resolution approach, if the search window is still centered on the current block and not up-scaled, the computed motion vector might correspond to a local minimum of the cost function used. In [5], Hsiao *et al.* exploit the spatial/temporal correlation in the motion vector fields to predict the initial search point for the long-term motion estimation. This initial search point accelerates the convergence speed of fast search algorithms for motion estimation.

In the following section, we present the multi-resolution motion estimation method based on tubes for HD sequences. In section 3, we describe the Nelder-Mead Simplex method. In section 4, we combine both methods to get an efficient long-term motion estimation. Finally, we show the simulation results in section 5 and give a conclusion in section 6.

2. MULTI-RESOLUTION MOTION ESTIMATION BASED ON TUBES

In [6], Pechard *et al.* proposed a multi-resolution motion estimation in order to accelerate motion estimation with HD frames. We describe this method because it gives the concept of spatio-temporal tubes.

[1]This research was carried out within the framework of the ArchiPEG project financed by the ANR (convention N°ANR05RIAM01401).

The HD frames are spatially filtered and sub-sampled by a factor of six. First, the frames are sub-sampled by a factor of two, and then, by a factor of three. Before each sub-sampling step, an adequate (half-bandwidth, and then, one-third the bandwidth) low-pass filter is performed to avoid aliasing.

From the filtered and spatially sub-sampled frames, the motion estimation is computed. We use five consecutive frames, and consider a uniform motion between the frames. Thus, we create a tube between the frames. Each block is simultaneously compared to its potentially corresponding aligned blocks in the two previous frames and in the two next frames, as illustrated in Fig. 1. The global error, MSE_G, is obtained by the sum of the four Mean Square Error (MSE) between the current block and its corresponding aligned blocks in the two previous frames and in the two next frames, as written in Eq. 1.

$$MSE_G = \sum_k MSE_k, \;\; k = -2, -1, +1, +2. \tag{1}$$

MSE_k takes into account the three YUV components of each block, as written in Eq. 2.

$$\begin{aligned} MSE_k = \;\; & \tfrac{1}{N \times N} \Big(\textstyle\sum_{i,j=0}^{N-1} (C_Y(i,j) - Rk_Y(i+\lambda_k.m, j+\lambda_k.n))^2 \\ & + \textstyle\sum_{i,j=0}^{N-1} (C_U(i,j) - Rk_U(i+\lambda_k.m, j+\lambda_k.n))^2 \\ & + \textstyle\sum_{i,j=0}^{N-1} (C_V(i,j) - Rk_V(i+\lambda_k.m, j+\lambda_k.n))^2 \Big), \end{aligned} \tag{2}$$

with $\lambda_0 = 2$, $\lambda_1 = 1$, $\lambda_3 = -1$, and $\lambda_4 = -2$, and (m, n) the motion vector between the current frame F_t and the immediately previous frame F_{t-1}. C_Y, C_U, C_V, Rk_Y, Rk_U, and Rk_V represent respectively the three YUV components of the current frame, and of the frames used as reference for motion estimation with blocks of size $N \times N$.

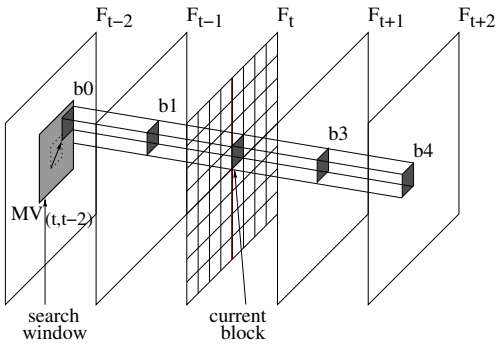

Fig. 1. The five frames used to determine the motion vector of a given block. A spatio-temporal tube is obtained.

The motion vector of a block from F_t is chosen such that it gives the lowest MSE_G between the current block and its corresponding blocks in the surrounding four frames. The motion vectors are estimated at the lowest resolution, and then, they are up-scaled appropriately to the higher resolution to be used as an initial search point.

The alignment of the blocks induces a spatio-temporal constraint in the tubes computation. This constraint produces very smoo-th and coherent motion vectors fields for the HD video sequences. This propriety is fundamental because the tubes motion vectors will be used to initialize the long-term motion estimation (LTME).

3. NELDER-MEAD SIMPLEX METHOD

The Nelder-Mead Simplex (NMS) method [7] attempts to minimize a scalar-valued nonlinear function of n real variables using only function values, without any derivative information (explicit or implicit). The Nelder-Mead algorithm applied to strictly convex functions in one or two dimensions converges [8], so we heve adapted this method for long-term motion estimation.

The Nelder-Mead algorithm minimizes a real-valued function $f(\mathbf{x})$ for $\mathbf{x} \in \Re^n$. To define a complete NMS method, four scalar parameters must be specified : coefficients of *reflection* (ρ), *expansion* (χ), *contraction* (γ), and *shrinkage* (σ). According, to the original NMS method, these four parameters should satisfy : $\rho > 0$, $\chi > 1$, $\chi > \rho$, $0 < \gamma < 1$, and $0 < \sigma < 1$. We use the following values for the parameters (the nearly universal choices in the standard NMS), $\rho = 1$, $\chi = 2$, $\gamma = \frac{1}{2}$, and $\sigma = \frac{1}{2}$.

At the k^{th} iteration, $k \geq 0$, a non degenerate simplex Δ_k is given with its $n+1$ vertices, each of which is a point in $\Re^n$. These vertices, labeled $\mathbf{x}_1^k, ..., \mathbf{x}_{n+1}^k$, are ordered in the following way,

$$f(\mathbf{x}_1^{(k)}) \leq f(\mathbf{x}_2^{(k)}) \leq ... \leq f(\mathbf{x}_{n+1}^{(k)}).$$

The k^{th} iteration generates a set of $N+1$ vertices which defines a different simplex for the next iteration, so $\Delta_{k+1} \neq \Delta_k$. As we want to minimize f, $\mathbf{x}_1^{(k)}$, $\mathbf{x}_n^{(k)}$ and $\mathbf{x}_{n+1}^{(k)}$ are respectively the best point, the n^{th} best point, and the worst point.

As we use the NMS method for block matching, the function f, that we want to minimize is the MSE. Block matching is a two dimensions problem, so $n = 2$, and the $n + 1 = 3$ vertices of the simplex represent the positions of candidate blocks.

One iteration of the NMS Algorithm can be described as the following way :

1. Order the 3 vertices to satisfy $f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq f(\mathbf{x}_3)$.
2. Compute the ***reflection point*** $\mathbf{x}_r$ from

   $$\mathbf{x}_r = \bar{\mathbf{x}} + \rho(\bar{\mathbf{x}} - \mathbf{x}_3) = (1+\rho)\bar{\mathbf{x}} - \rho\mathbf{x}_3,$$

   where $\bar{\mathbf{x}} = \sum_{i=1}^{2} \mathbf{x}_i / 2$ is the centroid of the 2 best points. If $f(\mathbf{x}_1) \leq f(\mathbf{x}_r) < f(\mathbf{x}_2)$, accept the reflected point $\mathbf{x}_r$ and terminate the iteration.
3. If $f(\mathbf{x}_r) < f(\mathbf{x}_1)$, calculate the ***expansion point*** $\mathbf{x}_e$,

   $$\mathbf{x}_e = \bar{\mathbf{x}} + \chi(\mathbf{x}_r - \bar{\mathbf{x}}) = (1 + \rho\chi)\bar{\mathbf{x}} - \rho\chi\mathbf{x}_3,$$

   if $f(\mathbf{x}_e) < f(\mathbf{x}_r)$, accept $\mathbf{x}_e$ and terminate the iteration; otherwise accept $\mathbf{x}_r$ and terminate the iteration.

4. If $f(\mathbf{x}_r) \geq f(\mathbf{x}_2)$, perform a ***contraction*** between $\bar{\mathbf{x}}$ and the better of $\mathbf{x}_3$ and $\mathbf{x}_r$

   - a) If $f(\mathbf{x}_2) \leq f(\mathbf{x}_r) < f(\mathbf{x}_3)$, perform an ***outside contraction***,

     $$\mathbf{x}_c = \bar{\mathbf{x}} + \gamma(\mathbf{x}_r - \bar{\mathbf{x}}) = (1 + \rho\gamma)\bar{\mathbf{x}} - \rho\gamma\mathbf{x}_3,$$

     if $f(\mathbf{x}_c) \leq f(\mathbf{x}_r)$, accept $\mathbf{x}_c$ and terminate the iteration; otherwise go to step 5.
   - b) If $f(\mathbf{x}_r) \geq f(\mathbf{x}_3)$, perform an ***inside contraction***,

     $$\mathbf{x}_{cc} = \bar{\mathbf{x}} + \gamma(\bar{\mathbf{x}} - \mathbf{x}_3) = (1 - \gamma)\bar{\mathbf{x}} + \gamma\mathbf{x}_3),$$

     if $f(\mathbf{x}_{cc}) < f(\mathbf{x}_3)$, accept $\mathbf{x}_{cc}$ and terminate the iteration; otherwise go to step 5.

5. Evaluate f at the 2 points $\mathbf{v}_i = \mathbf{x}_1 + \sigma(\mathbf{x}_i - \mathbf{x}_1)$, $i = 2, 3$. The unordered vertices simplex at the next iteration consist of $\mathbf{x}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ (***shrinkage***).

Figure 2 shows the effects of reflection, expansion, contraction and shrinkage.

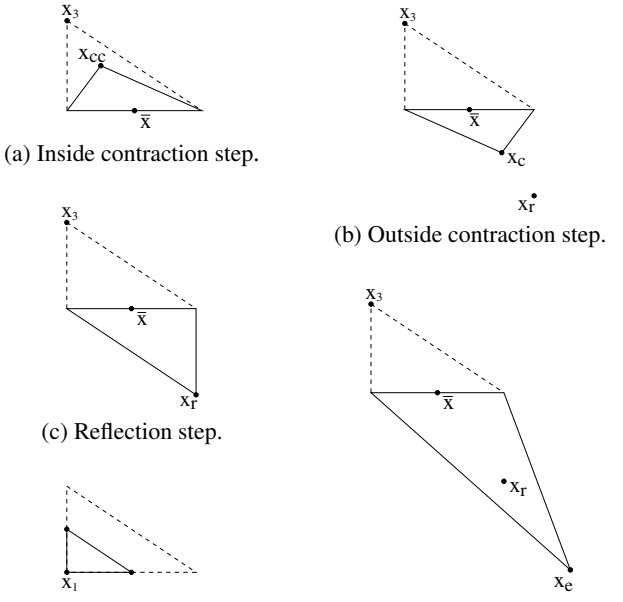

Fig. 2. Nelder-Mead simplices after different steps. The original simplex is shown with a dashed line.

4. LONG-TERM MOTION ESTIMATION

Our goal is now to improve the long-term motion estimation using both fast methods : the spatio-temporal tube to find an optimal initial search point, and the NMS method to achieve the long-term motion estimation.

If the reference frame, used for motion estimation of the current frame, is the immediately previous frame, these two frames are highly temporally correlated and the motion vector will remain weak. Thus, the search window does not need to be too large and is centered on the current block to be estimated. But, if the reference frame is temporally far from the current one, the spatial correlation decreases. In this case, it is important to predict a good initial search point for motion estimation that will not obtain a local minimum of the cost function and reduces the calculation time.

Considering the current frame F_t, our strategy is to first use the motion vector of the tube to get an optimal initial search point for the long-term motion estimation. This motion estimation is realized using the multi-resolution motion estimation method explained in section 2. The motion vector obtained, $MV_{(t,t-1)}$, is then up-scaled by the time interval between the current frame and the long-term reference (see Eq. 3).

$$MV_{pred(t,t-k)} = k \times MV_{(t,t-1)}. \quad (3)$$

The predicted motion vector, $MV_{pred(t,t-k)}$ is used as the initial search point for long-term motion estimation, as illustrated in Fig. 3. For the long-term motion estimation, we use the NMS method described in section 3 to refine the motion vector.

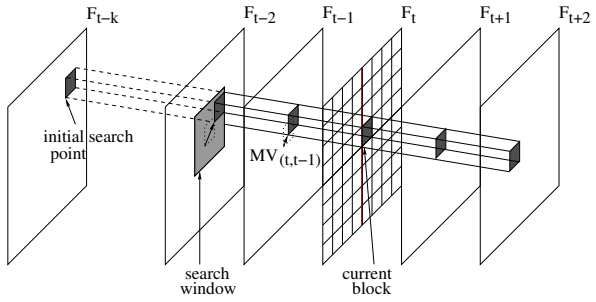

Fig. 3. Initialization of the long-term motion estimation.

5. SIMULATION RESULTS

We used 1080p HD sequences (*blue_sky*, *pedestrian*, *station*, *rush_hour*) from SVT [9]. These videos can be classified in two categories according to their motion and their content. The first category with high motion contains the two sequences *pedestrian*, and *rush_hour*. The other category contains sequences with slow motion like *blue_sky*, and *station*. For the motion estimation the blocks size is about 16×16. The best motion vector is obtained by a search on integer-pel positions with a search window of size 20×20 (± 20 pixels in each direction) for the Full Search method. Table 1 and Table 2 show the comparison results in term of computing times and PSNR, between our long-term motion estimation (LTME) method, a Full Search (FS) method centered on the block, and a Full Search method with an initial search point (FSI). This initial search point is obtained using a Full Search between the current frame and the immediately previous frame, the motion vector obtained is then extrapolated to the long term reference frame using equation 3. We want to show that our

method is faster, and more accurate than a FS method but also performs almost as well as the FSI. The PSNR is computed between the current frame F_t and the frame F_{t-k} that is motion compensated using the computed long-term vectors. The encoding speed has been measured as the relative variation in computation time (see Eq. 4) between the considered methods (T) and the FS method (T_{FS}):

$$\triangle T(\%) = \frac{T - T_{FS}}{T_{FS}} \times 100. \quad (4)$$

According to tables 1 and 2, our method can accelerate the motion estimation up to 46% and performs better than Full Search centered on the current block. If the video contains important and irregular motions (*pedestrian* and *rush_hour*) our LTME method performs better in terms of PSNR and computing time than the FSI. Indeed with the NMS algorithm, the motion estimation is not forced by a search window and can find a better distant block.

Sequence	FS		LTME		FSI	
	PSNR	ΔT(%)	PSNR	ΔT(%)	PSNR	ΔT(%)
pedestrian	21.54	-	27.05	−28.55	23.91	99.36
rush_hour	27.67	-	30.65	−27.20	30.36	98.70
station	31.93	-	34.24	−46.00	36.63	98.22
blue_sky	22.48	-	28.80	−42.37	31.61	98.50

Table 1. Comparison of PSNR (dB) and coding speed with a long time interval of **five** frames between F_t and F_{t-k}.

Sequence	FS		LTME		FSI	
	PSNR	ΔT(%)	PSNR	ΔT(%)	PSNR	ΔT(%)
pedestrian	19.37	-	25.08	−27.95	20.79	96.89
rush_hour	23.54	-	27.94	−26.87	26.00	96.94
station	28.58	-	31.04	−46.14	34.16	92.63
blue_sky	20.14	-	24.56	−42.82	28.92	96.30

Table 2. Comparison of PSNR (dB) and coding speed with a long time interval of **ten** frames between F_t and F_{t-k}.

As illustrated in figure 4, the simulations results show, that we obtained better compensated frames (PSNR) with our LTME when the video contains important motions, whatever the time interval between the current frame and the long-term frame used as reference.

6. CONCLUSIONS

In this paper, we presented a new fast long-term motion estimation (LTME) method for HD sequences. First, a multi-resolution motion estimation is realized in order to obtain a tube as an initial search point for the long-term reference. Then, we use the Nelder-Mead Simplex (NMS) method to refine the motion vectors between the current and long-term reference frames. The results obtained with our LTME are better in terms of computing time and PSNR compared to a Full Search method. We also compare our LTME with a Full Search method using a initial search point obtained from a full search on the previous frame and extrapolated to the long-term reference frame. In case of important and irregular motions in the video sequence, our LTME obtains more accurate results owing to the the NMS algorithm which is not limited by a search window and can search for a better distant point.

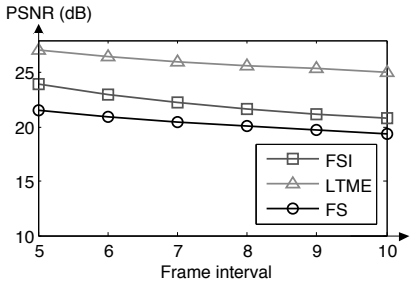

Fig. 4. PSNR comparisons for the different methods with different time intervals for the sequence *pedestrian*.

7. REFERENCES

[1] I. E. G. Richardson, *H.264 and MPEG-4 video compression: Video Coding for Next-Generation Multimedia.*, Chippenham, September 2003.

[2] B. D. Choi, M. C. Hwang, J. K. Cho, J. S. Kim, J. H. Kim, and S. J. Ko, "Realtime H.264 Encoding System using Fast Motion Estimation and Mode Decision," *Lecture Notes in Computer Science*, vol. 3824, pp. 174 – 183, December 2005.

[3] Z. Liu, Y. Song, T. Ikenaga, and S. Goto, "Low-Pass Filter Based VLSI Oriented Variable Block Size Motion Estimation Algorithm for H.264," Toulouse, France, in Proc. ICASSP 2006.

[4] T. Wiegand, X. Zhang, and B. Girod, "Long-Term Memory Motion-Compensated Prediction," IEEE Trans. on Circuits and Systems for Video Technology, 1999, vol. 9, pp. 70 – 84.

[5] Y. H. Hsiao, T. H. Lee, and P. C. Chang, "Short/Long-Term Motion Vector Prediction in Multi-Frame Video Coding System," Singapore, in Proc. ICIP 2004.

[6] S. Péchard, P. Le Callet, M. Carnec, and D. Barba, "A new methodology to estimate the impact of H.264 artefacts on subjective video quality," Scottsdale, VPQM 2007.

[7] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308 – 313, 1965.

[8] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence Properties of the Nelder–Mead Simplex Method in Low Dimensions," *SIAM J. on Optimization*, vol. 9, no. 1, pp. 112 – 147, 1998.

[9] SVT, "Overall-quality assessment when targeting wide xga flat panel displays," Tech. Rep., SVT corporate development technology, 2002.